Proceeding, Seminar Ilmiah Nasional Komputer dan Sistem Intelijen (KOMMIT 2008)
Auditorium Universitas Gunadarma, Depok, 20-21 Agustus 2008 ISSN : 1411-6286

PEMBUATAN APLIKASI UNTUK MENDESAIN BASIS DATA BERDASARKAN SEMANTIC OBJECT MODEL

[1]Oviliani Yenty Yuliana
[2]Silvia Rostianingsih

Jurusan Teknik Informatika Universitas Kristen Petra
Jl. Siwalan Kerto 121-131 Surabaya 60236
([1]ovi@peter.petra.ac.id, [2]silvia@peter.petra.ac.id)

ABSTRAK

Penelitian yang dilakukan bertujuan untuk membuat aplikasi yang dapat digunakan untuk mendesain model data dengan pendekatan Semantic Object Model (SOM). Aplikasi tersebut juga dilengkapi dengan fasilitas untuk mentransformasikan SOM ke dalam Basis Data Oracle. Pembuatan aplikasi mengikuti Metode Software Developement Life cycle (SDLC). Berdasarkan hasil pengujian, Aplikasi SOM dapat digunakan untuk mendesain Model Data SOM dan mentransformasikan model data tersebut ke dalam basis data Oracle.

Kata Kunci: Model Data, Semantic Object Model, Meta Schema, Meta Tabel, Basis Data

1. PENDAHULUAN

SOM adalah model data dengan paradigma *object*. SOM dirancang untuk menangkap kebutuhan aplikasi yang lebih baik dan menyediakan struktur data untuk aplikasi basis data yang lebih kaya (Codd:1979, Hull:1987). Hal tersebut dikarenakan SOM menyediakan sejumlah mekanisme yang mewakili keterkaitan antara data yang kompleks dan terstruktur untuk pembuatan aplikasi (Hammer:1981, Peckham:1988). Dalam penelitian ini menggunakan simbol dan mekanisme SOM yang diusulkan oleh Kroenke (2006).

Terdapat beberapa Aplikasi SOM, seperti: Salsa dan Tabledesigner. Namun aplikasi tersebut hanya menyediakan fasilitas untuk mentranformasikan Model Data SOM ke dalam basis data Microsoft Access. Selain itu, akhir-akhir ini aplikasi tersebut sulit diperoleh karena semua *home page* yang menyediakan aplikasi tersebut sudah tidak aktif. Kebutuhan Aplikasi SOM akan semakin meningkat seiring dengan meningkatnya kebutuhan rancang sistem dan pemrograman yang berorientasi obyek.

Untuk itu, penelitian yang dilakukan bertujuan untuk membuat aplikasi yang dapat digunakan untuk mendesain Model Data SOM. Selain itu aplikasi yang dibuat dilengkapi dengan fasilitas untuk mentransformasikan Model Data SOM ke dalam Basis Data Oracle.

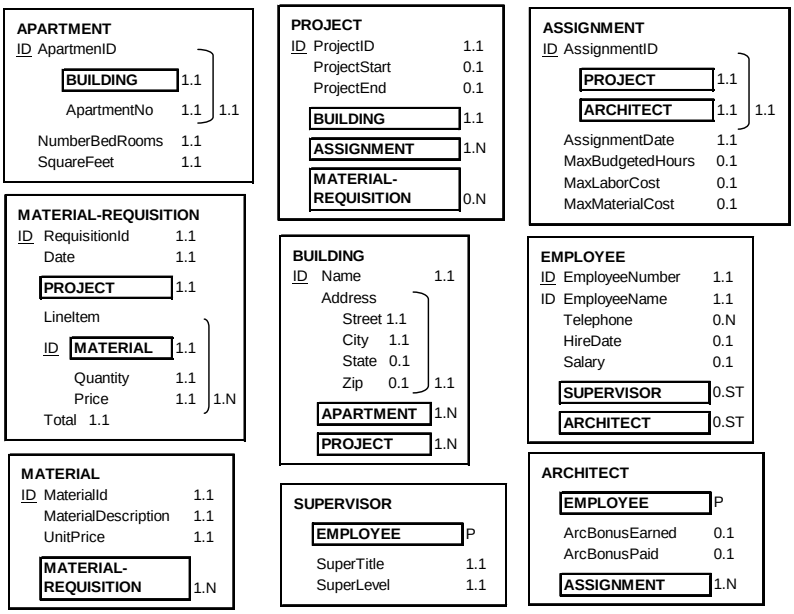
2. TINJAUAN PUSTAKA

Kroenke (2006) menyatakan *Semantic Object* (SO) memiliki 3 jenis atribut, yaitu: *simple*, *group*, dan *object*. Atribut *Simple* dan *Group* dikenal dengan *non object attribute*. Atribut *Simple* adalah atribut tunggal, misalnya Atribut Name dalam Obyek BUILDING pada Gambar 1. Atribut *Group* adalah atribut yang berisi atribut lain, sebagai contoh Atribut Address dalam Obyek BUILDING. Atribut tersebut berisi atribut: Street, City, State, dan Zip. Sedangkan Atribut *Object* adalah atribut yang berfungsi sebagai penghubung antara obyek, misalnya Atribut APARTEMENT dalam Obyek BUILDING. Atribut APARTEMENT berfungsi untuk menghubungkan Obyek BUILDING dengan Obyek APARTEMENT.

SO memiliki dua jenis *identifier*, yakni: *Unique* (ID) dan *Non Unique* (ID). Sebagai contoh dalam Obyek EMPLOYEE, *Unique Identifier* adalah EmployeeNumber dan *Non Unique Identifier* adalah EmployeeName. *Group Identifier* adalah *identifier* yang merupakan gabungan dari beberapa atribut. Sebagi contoh *Unique Identifier* Obyek APARTMENT adalah {Name, ApartmentNo}.

Cardinality SO dinyatakan dengan format m.n, m untuk minimum *cardinality* dan n untuk maksimum *cardinality*. Terdapat dua istilah dalam maksimum *cardinality*: *single value* dan *multi value*. *Single value* untuk atribut dengan maksimum *cardinality* 1. Sedangkan *multi value* untuk atribut dengan maksimum *cardinality* bisa lebih dari 1. Sebagai contoh: Atribut EmployeeNumber dalam Obyek EMPLOYEE minimum dan maksimum *cardinality* adalah 1. Maksudnya Atribut EmployeeNumber harus berisi data. Sedangkan Atribut HireDate minimum *cardinality* adalah 0 dan maksimum *cardinality* adalah 1. Atribut HireDate boleh tidak diisi data. Contoh lainnya, Atribut Telephone minimum *cardinality* adalah 0 dan maksimum *cardinality* adalah N. Artinya Employee dapat tidak memiliki Telephone atau memiliki Telephone lebih dari satu.

Kroenke (2006) mengusulkan 7 tipe Obyek SO, yaitu: *simple*, *composite*, *archetype*/*version*, *parent*/*subtype*, *compound*, *association*, dan *hybrid*. Ketujuh tipe obyek digunakan untuk administrasi proyek dapat dilihat pada Gambar 1. Obyek *Simple* adalah obyek yang hanya berisi *single value* dan *non object attribute*. Obyek *Composite* adalah obyek yang berisi satu atau beberapa *multi value non object* atribut, contoh Atribut Telephone dalam Obyek BUILDING. Obyek *Archetype*/*Version* menghasilkan versi SO, dengan ciri *identifier* berupa Atribut *Group*, salah satu dari Atribut *Group* berupa Atribut *Object*. Contoh *Group Identifier* ApartmenID pada Obyek APARTMENT berisi Atribut *Object* BUILDING dan Atribut *Simple* ApartmentNo. Jadi *Identifier* Obyek APARTMENT adalah {Name, ApartmentNo}. Obyek *Parent*/*Subtype* memiliki karakteristik *inheritance*, maksudnya Obyek *Subtype* mewarisi semua Atribut *Parent*. Selain itu, Obyek *Parent* memiliki maksimum *cardinality* ST, yang menunjukkan *parent* tersebut memiliki *subtype* apa saja. Sedangkan dalam Obyek *Subtype* memiliki Atribut *Object* yang ditandai oleh ”P”. Contohnya, EMPLOYEE adalah Obyek *Parent* dengan 2 *subtype*: SUPERVISOR dan ARCHITECT. Obyek *Compound* adalah obyek yang berisi satu atau beberapa Atribut *Object*. Sebagai contoh dalam Obyek PROJECT terdapat Atribut *Object* APARTMENT dan sebaliknya dalam Obyek APARTMENT terdapat Atribut *Object* PROJECT. Ada 3 jenis hubungan antara obyek, yaitu: *one-to-one*, *one-to-many*, dan *many-to-many*. Hubungan tersebut dapat dilihat dari maksimum *cardinality* dari kedua Atribut *Object*. Obyek *Association* menghubungkan dua atau lebih obyek dan menyimpan data yang menunjukkan hubungan antara obyek tersebut. Karakteristik dari Obyek *Association* adalah *identifier* berupa Atribut *Object* dalam Atribut *Group*. *Identifier* selain berfungsi untuk mengunikkan data, juga berfungsi untuk menghubungkan antara obyek. Sebagai contoh, *identifier* pada ASSIGNMENT berisi Atribut *Object* PROJECT dan ARCHITECT. Obyek *Hybrid* adalah obyek yang memiliki satu atau beberapa *multi value* Atribut *Group*, dimana dalam *multi value* Atribut *Group* tersebut berisi 1 Atribut *Object*, contohnya Obyek MATERIAL_REQUISITION.

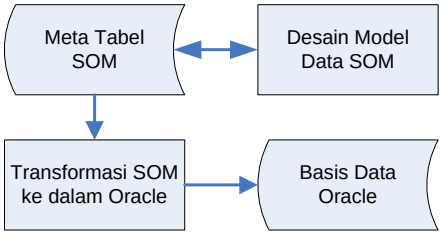


Gambar 1. SOM untuk administrasi proyek
Sumber: Yuliana (2007)

3. METODE PENELITIAN

Pembuatan Aplikasi SOM mengikuti 5 langkah awal metode SDLC yang diusulkan oleh Galin (2004).

- Mendefinisikan kebutuhan, seperti yang diungkap dalam pendahuluan bahwa diperlukan adanya aplikasi yang dapat digunakan untuk menunjang desain basis data dengan penekatan SOM.
- Analisa: pada tahap ini dilakukan analisa komponen SOM untuk pembuatan model data SOM dan dianalisa mekasnisme transformasi untuk ketujuh tipe obyek SOM.
- Perancangan: pada tahap ini dirancang meta *schema* dan meta tabel untuk menunjang desain basis data dengan pendekatan SOM, dibuat algoritma, dan dirancang tampilan layar.



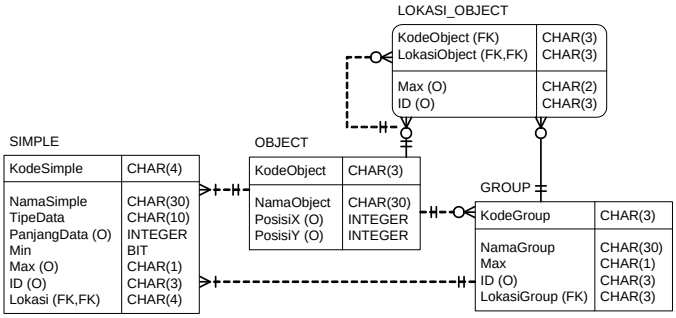
Gambar 2. Blok diagram rancangan Aplikasi SOM

- Pembuatan program: program dibuat dengan bahasa pemrograman Borland Delphi.
- Pengujian Sistem: dilakukan pengujian terhadap Aplikasi SOM agar dapat digunakan untuk menggambar model data SOM dan mentransformasikannya ke dalam basis data Oracle. Kesimpulan diambil berdasarkan hasil pengujian.

4. HASIL DAN PEMBAHASAN

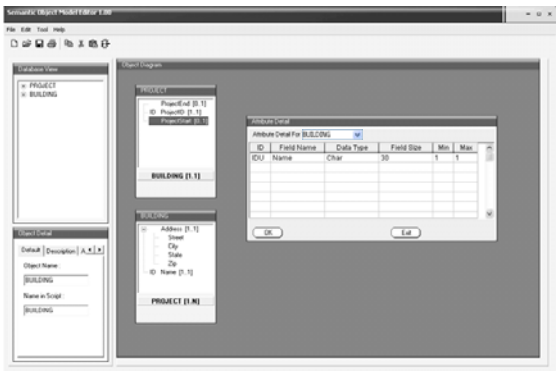
Untuk dapat menunjang pembuatan Model Data SOM seperti yang tampak pada Gambar 1, maka diperlukan meta tabel dalam aplikasi. Rancangan meta *schema* untuk meta tabel tampak pada Gambar 3. Nama obyek, Atribute *Simple*, dan Atribute *Group* masing-masing disimpan dalam *Entity* OBJECT, SIMPLE, dan GROUP. Pada Gambar 1, tampak Atribut *Simple* dan *Object* dapat berada dalam *Object* atau *Group*. Untuk itu *Entity* SIMPLE dan GROUP perlu mencantumkan lokasi atribut tersebut berada di mana. Selain itu, suatu Atribut *Object* dimungkinkan untuk berada dalam beberapa obyek, misalnya Atribut *Object* ARCHITECT terdapat dalam Obyek ASSIGNMENT dan EMPLOYEE. Untuk

keperluan tersebut dicatat dalam *Entity* LOKASI_OBJECT.



Gambar 3. Meta *schema* aplikasi SOM

Desain layar untuk membuat Model Data SOM ditampilkan pada Gambar 4. Setiap obyek dicatat pada tabel *Object*. Data obyek yang disimpan adalah nama, posisi sumbu X, dan posisi sumbu Y. Kode obyek diisi secara otomatis oleh aplikasi. Atribut *Simple* dapat berada dalam Obyek atau Atribut *Group*. Untuk itu, saat menambah Atribut *Simple* harus ditetapkan berada di mana.



Gambar 4. Aplikasi SOM

Berdasarkan Gambar 4, tampak kemungkinan data pada kolom ID adalah spasi, ID (*Non Unique Identifier*), dan IDU (*Unique Identifier*). Pilihan data untuk kolom Min adalah 0 atau 1, dengan nilai awal 0 dan secara otomatis diisi 1 kalau kolom ID diisi IDU. Kemungkinan data untuk kolom Max adalah 1, N, P, atau ST.

Secara umum basis data Oracle terbentuk dengan mentransformasikan meta tabel SOM. Proses Model Data SOM dilakukan dengan mentransformasikan setiap obyek dalam Tabel OBJECT menjadi satu tabel. Setiap Atribut *Simple* yang berada dalam obyek dan Atribut *Group* dijadikan kolom, tergantung dimana atribut tersebut berada. Selain itu *Unique Identifier* dari Atribut *Object* dijadikan kolom pada obyek dimana atribut tersebut berada.

```
{Procedur untuk menambahkan kolom baru pada suatu tabel}
Procedure Create_Field(C_Simple)
    For R_Simple in C_Simple Loop
        Create Field(R_Simple.NamaSimple)
        Set Property Field(R_Simple.TipeData, R_Simple.PanjangData)
        If R_Simple.Min=0 Then Set Property Field Null
        Else Set Property Field Not Null
    EndLoop
EndProcedure

Do While Not EOF(OBJECT)
    Create Table(NamaObject)
    {Setiap simple attribute dari tabel SIMPLE dijadikan kolom baru}
    Cursor C_Simple Is
        Select NamaSimple, TipeData, PanjangData, Min From SIMPLE
            Where SIMPLE.Object=OBJECT.KodeObject and Max="1" and Left(Object,1)="O"
        Create_Field(C_Simpel)
    {Setiap simple attribute dalam group attribute dijadikan kolom baru}
    Cursor C_Group Is
```

```
            Select KodeGroup From GROUP
                Where GROUP.LokasiGroup=OBJECT.KodeObject and Max="1"
        For R_Group In C_Group Loop
            Cursor C_Simple Is
                Select NamaSimple, TipeData, PanjangData, Min From SIMPLE
                    Where SIMPLE.Object=R_Group.KodeGroup and Max="1"
            Create_Field(C_Simple)
            Cursor C_Lokasi Is
                Select KodeObject From LOKASI_OBJECT
                    Where LOKASI_OBJECT.LokasiObject=R_Group.KodeGroup and Max="1"
            For R_Lokasi In C_Lokasi Loop
                Cursor C_Simple Is
                    Select NamaSimple, TipeData, PanjangData, Min From SIMPLE
                        Where SIMPLE.Object=R_Lokasi.KodeObject and ID="IDU"
                Create_Field(C_Simple)
        EndLoop
        {Setiap identifier suatu object dijadikan kolom baru}
        Cursor C_Lokasi Is
            Select KodeObject From LOKASI_OBJECT
                Where LOKASI_OBJECT.LokasiObject=OBJECT.KodeObject and ((Max="1") or (Max="P"))
        For R_Lokasi In C_Lokasi Loop
            Cursor C_Simple Is
                Select NamaSimple, TipeData, PanjangData, Min From SIMPLE
                    Where SIMPLE.Object=R_Lokasi.KodeObject and ID="IDU"
            Create_Field(C_Simple)
        EndLoop
        Read Next Record OBJECT
EndDo
```

Gambar 5. Modul pentransformasi obyek dan atribut

```
    Cursor C_Group Is
        Select KodeGroup, NamaGroup, LokasiGroup From GROUP
            Where Max="N"
    For R_Group In C_Group Loop
        Create Table(NamaGroup)
        Cursor C_Simple Is
            Select NamaSimple, TipeData, PanjangData, Min From SIMPLE
                Where SIMPLE.Object=R_Group.KodeGroup and Max="1"
        Create_Field(C_Simple)
        Cursor C_Lokasi Is
            Select KodeObject From LOKASI_OBJECT
                Where R_Group.KodeGroup=LOKASI_OBJECT.LokasiObject and Max="1"
        For R_Lokasi In C_Lokasi Loop
            Cursor C_Simple Is
                Select NamaSimple, TipeData, PanjangData, Min From SIMPLE
                    Where SIMPLE.Object=R_Lokasi.KodeObject and ID="IDU"
        Create_Field(C_Simple)
    EndLoop
```

Gambar 6. Modul pentransformasi *multivalue* atribut *group*

```
    Cursor C_Simple1 Is
        Select NamaSimple, TipeData, PanjangData, Min, Object From SIMPLE
            Where Max="N"
    For R_Simple1 In C_Simple1 Loop
        Create Table(NamaSimple)
        Cursor C_Simple2 Is
            Select NamaSimple, TipeData, PanjangData, Min From SIMPLE
                Where R_Simple1.Object=SIMPLE.Object and ID="IDU"
        Create_Field(C_Simple2)
    EndLoop
```

Gambar 7. Modul pentransformasi *multivalue* atribut *simple*

Pembuatan *Single Primary Key* yang berasal dari Atribut *Simpel* (sebagai contoh ProjectID dalam Obyek PROJECT pada Gambar 1) dapat dilihat pada Gambar 8. Pembuatan *Composite Primary Key* yang berasal dari *identifier* berupa Atribut *Group* dapat dilihat pada Gambar 9. Pada Gambar 1 terdapat 2 *group identifier*, yakni: ApartmenID (berisi Atribut *Object* BUILDING dan Atribut *Simple* AprtmentNo) dan AssignmentID (keduanya adalah Atribut *Object*, PROJECT dan ARCHITECT). Pembuatan *Primary Key* untuk *multi value* Atribut

Group, sebagai contoh LineItem dalam Obyek MATERIAL-REQUISITION di Gambar 1, dilakukan dengan cara yang berbeda meskipun *Primary Key* berupa *Composite*. *Primary Key* untuk Tabel LineItem berupa gabungan dari *Primary Key* Obyek MATERIAL-REQUISITION (tempat Atribut *Group* tersebut berada) dan *Primary Key* dari Obyek MATERIAL. Jadi RequisitionID dan MaterialId dijadikan *Primary Key*. Proses pembuatan *Primary Key* secara detail dapat dilihat pada Gambar 10.

```
Cursor C_Simple Is
    Select NamaSimple, NamaObject From SIMPLE S, OBJECT O
        Where S.Object=O.KodeObject and ID=”IDU”
For R_Simple In C_Simple Loop
    Set Primary Key (R_Simple.NamaSimple) Into (R_Simple.NamaObject)
EndLoop
```

Gambar 8. Modul pembuat *primary key* (*single key*)

Untuk merelasikan antara tabel perlu ditetapkan *referential integrity*. Cara menetapkan *referential integrity* untuk tipe Obyek *Parent/Subtype* (kolom Max berisi P pada Tabel LOKASI_OBJECT) dapat dilihat pada Gambar 11. Sedangkan cara untuk mendefinisikan *referential integrity* antara tabel secara umum (kolom Max berisi N) dapat dilihat pada Gambar 12. Pendefinisian *referential integrity* untuk tabel yang terbentuk dari *multi value* Atribut *Group* dan *multi value* Atribut *Simpe* masing-masing secara berturut-turut dapat dilihat pada Gambar 13 dan Gambar 14.

```
Cursor C_Group Is
    Select KodeGroup, NamaObject From OBJECT O, GROUP G
        Where O.KodeObject=G.LokasiGroup and ID=”IDU”
For R_ Group In C_Group Loop
    {Salah satu key adalah object attribute}
    Cursor C_Lokasi Is
        Select KodeObject From LOKASI_OBJECT
            Where R_ Group.KodeGroup=LOKASI_OBJECT.LokasiObject
    VKey:=””
    For R_Lokasi In C_Lokasi Loop
        Select NamaSimple Into Tkey From SIMPLE
        Where SIMPLE.Object=R_Lokasi.KodeObject and ID=”IDU”
        VKey:=VKey+TKey
    EndLoop
    {Salah satu key adalah simple attribute}
    Cursor C_Simple Is
        Select NamaSimple From SIMPLE
            Where SIMPLE.Object=R_Group.KodeGroup
    For R_Simple In C_Simple Loop
        VKey:=Vkey+R_Simple.NamaSimple
    EndLoop
    Set Primary Key (VKey) Into (R_Group.NamaObject)
EndLoop
```

Gambar 9. Modul pembuat *primary key* (*composite key*)

```
Cursor C_Lokasi Is
    Select KodeObject, NamaGroup, LokasiGroup From GROUP G, LOKASI_OBJECT L
        Where L.LokasiObject=G.KodeGroup and G.Max<>”1” and L.ID=”IDU”
Vkey:=””
For R_Lokasi In C_Lokasi Loop
    Select NamaSimple Into Tkey From SIMPLE
        Where SIMPLE.Object=R_Lokasi.LokasiGroup and ID=”IDU”
    VKey:=VKey+TKey
    Select NamaSimple Into Tkey From SIMPLE
        Where SIMPLE.Object=R_Lokasi.KodeObject and ID=”IDU”
    VKey:=VKey+TKey
    Set Primary Key (VKey) Into (R_Lokasi.NamaGroup)
EndLoop
```

Gambar 10. Modul pembuat *primary key* tabel yang terbentuk dari *multivalue* atribut *group*

```
Cursor C_Lokasi Is
    Select KodeObject, LokasiObject, S.NamaSimple From SIMPLE S, LOKASI_OBJECT L
        Where L.KodeObject=S.Object and L.Max="P" and S.ID="IDU"
For R_Lokasi In C_Lokasi Loop
    Select NamaObject Into VParent From OBJECT
        Where R_Lokasi.KodeObject=OBJECT.KodeObject
    Select NamaObject Into VChild From OBJECT
        Where R_Lokasi.LokasiObject=OBJECT.KodeObject
    Set Referential Integrity R_Lokasi.NamaSimple From Vchild Into VParent
EndLoop
```

Gambar 11. Modul pembuat *referential integrity* untuk obyek *parent*/*subtype*

```
Cursor C_Lokasi Is
    Select KodeObject, LokasiObject From LOKASI_OBJECT
        Where Max="N" and Not Exists(Select * From GROUP Where Max="N")
For R_Lokasi In C_Lokasi Loop
    Select NamaSimple, NamaObject Into VFKey, VParent From SIMPLE S, OBJECT O
        Where O.KodeObject=R_Lokasi.LokasiObject and S.Object=R_Lokasi.LokasiObject and ID="IDU"
    Select NamaObject Into VChild From OBJECT
        Where R_Lokasi.LokasiObject=OBJECT.KodeObject
    Set Referential Integrity VFKey From Vchild Into VParent
EndLoop
```

Gambar 12. Modul pembuat *referential integrity* antara obyek

```
Cursor C_Group Is
    Select KodeObject, NamaGroup, LokasiGroup From GROUP G, LOKASI_OBJECT L
        Where L.LokasiObject=G.KodeGroup and G.Max="N"
For R_Group In C_Group Loop
    Select NamaSimple, NamaObject Into VFKey, VParent From SIMPLE S, GROUP G
        Where S.Object=R_Group.KodeObject and G.KodeGroup=R_Group.KodeObject and S.ID="IDU"
    Set Referential Integrity VFKey From NamaGroup Into VParent
    Select NamaSimple, NamaObject Into VFKey, VParent From SIMPLE S, GROUP G
        Where S.Object=R_Group.LokasiGroup and G.KodeGroup=R_Group.LokasiGroup and S.ID="IDU"
    Set Referential Integrity VFKey From NamaGroup Into VParent
EndLoop
```

Gambar 13. Modul pembuat *referential integrity* untuk tabel yang terbentuk dari *multivalue* atribut *group*

```
Cursor C_Simple Is
    Select Object, NamaSimple, NamaObject From SIMPLE S, OBJECT O
        Where S.Object=O.KodeObject and Max="N"
For R_Simple In C_Simple Loop
    Select NamaSimple Into VFKey From SIMPLE
        Where SIMPLE.Object=R_Simple.Object and ID="IDU"
    Set Referential Integrity VFKey From R_Simple.NamaSimple Into R_Simple.NamaObject
EndLoop
```

Gambar 14. Modul pembuat *referential integrity* untuk tabel yang terbentuk dari *multivalue* atribut *simple*

5. KESIMPULAN DAN SARAN

Aplikasi yang dihasilkan dapat membuat model data dengan pendekatan SOM. Selain itu, model data SOM dapat ditransformasikan ke dalam Basis Data Oracle. Pada aplikasi transformasi tidak memperhatikan *minimum constraint* dari obyek dan Atribut *Group*. Aplikasi SOM dapat dikembangkan untuk dapat menghasilkan file berupa *Structured Query Language* ANSI *script*. Dengan demikian luaran aplikasi yang dihasilkan dapat diimplementasikan ke dalam berbagai basis data. Selain itu, aplikasi dapat dikembangkan dengan mengimplementasikan keseluruhan *property* data termasuk *minimum constraint* dari Atribut *Object* dan *Group*.

DAFTAR PUSTAKA

[1] Codd, E. F. 1979. "Extending the Database Relational Model *to* Capture More Meaning." ACM Transactions on Database Systems. Volume 4, Nomor 4.

[2] Galin Daniel. 2004. *Software*

Quality Assurance: From Theory To Implementation 1st. Pearson Prentice Hall, New Jersey.

[3] Hammer, Michael and Dennis McLeod. 1981. "Database Description with SDM: A Semantic Database Model." ACM Transactions on Database Systems. Volume 6, Nomor 3.

[4] Hull, Richard and Roger King. 1987. "Semantic Database Modeling: Survey, Applications, and Research Issues." ACM Computing Surveys. Volume 19, Nomor 3.

[5] Kroenke, David M. 2006. *Database Processing: Fundamentals, Design, and Implementation 10th*. Pearson Prentice Hall, New Jersey.

[6] Peckham, Joan and Fred Maryanski. 1988. "Semantic Data Models." ACM Computing Surveys. Volume 20, Nomor 3.

[7] Tabledesigner. 13-12-2007. *Tabledesigner-Semantic Object Modeling*. http://coolstrategy.com/som.htm.

[8] Yuliana, Oviliani Y. 2007. "Semantic Object Model and Flat XML Schema." Proceeding of the 1st ICSIIT. Graha Ilmu, Jogyakarta.